Information Sciences 465 (2018) 144–161

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Efficient and effective influence maximization in social networks: A hybrid-approach

Yun-Yong Ko, Kyung-Jae Cho, Sang-Wook Kim*

Department of Computer and Software, Hanyang University, Seoul, Korea

ARTICLE INFO

Article history:
Received 19 April 2017
Revised 22 June 2018
Accepted 1 July 2018
Available online 10 July 2018

Keywords:
Social network
Information diffusion
Influence maximization
Monte-Carlo simulations

ABSTRACT

Influence Maximization (IM) is the problem of finding a seed set composed of k nodes that maximize their influence spread over a social network. Kempe et al. showed the problem to be NP-hard and proposed a greedy algorithm (referred to as *SimpleGreedy*) that guarantees 63% influence spread of its optimal solution. However, *SimpleGreedy* has two performance issues: at a micro level, it estimates the influence spread of a single node by running Monte-Carlo (MC) simulations that are fairly expensive; at a macro level, after selecting one seed at each step, it re-evaluates the influence spread of *every node* in a social network, leading to significant computational overhead. In this paper, we propose *Hybrid-IM* that addresses the two issues in both micro and macro levels by combining *PB-IM (Path Based Influence Maximization)* and *CB-IM (Community Based Influence Maximization)*. Furthermore, we identify two technical issues that could improve the performance of Hybrid-IM more and propose two strategies to address those issues. Through extensive experiments with four real-world datasets, we show that Hybrid-IM achieves great improvement (up to 43 times) in performance over state-of-the-art methods and finds the seed set that provides the influence spread very close to that of the state-of-the-art methods.

1. Introduction

The rapid growth of social network services (SNSs) such as Facebook and Twitter has greatly increased the number of people who use SNSs. SNSs serve as a medium through which people share their opinions and knowledge with others. Companies hope to spread the information related to themselves over a social network as much widely as possible by using a *viral marketing* strategy. Towards this goal, there have been a lot of research efforts to find such users who maximize this *word-of-mouth effect* in social networks [6,23,24,27,32,33,36].

This problem of *Influence Maximization (IM)* is defined as finding a seed set of k users who could maximize the influence spread over a social network [6,27]. Kempe et al. [15] showed that finding the optimal solution to the IM problem is *NP-hard* and proposed *SimpleGreedy* that selects the most influential node in each step and iterates this step k times to make the seed set have k nodes. The influence spread of a node is estimated by running the *Monte-Carlo (MC) simulations* 10,000 times and getting their average. Kempe et al. [15] demonstrated that the final seed set obtained by SimpleGreedy guarantees to provide over 63% influence spread of the optimal seed set. The influence spread of the seed set obtained by SimpleGreedy

* Corresponding author.
E-mail address: wook@hanyang.ac.kr (S.-W. Kim).

https://doi.org/10.1016/j.ins.2018.07.003
0020-0255/

is typically used as the ground truth for evaluating the influence spread of the seed set by other approximate or heuristic methods for IM in this field.

Although SimpleGreedy makes the IM problem solvable by approximation, it still takes too much time to select the seed set in practical sense by two performance issues: at a micro level, estimating the influence spread of a single node itself takes a significant amount of time due to running 10,000 MC-simulations; at a macro level, after selecting a new seed at a step, it needs to re-evaluate the influence spreads of every node in a social network.

Many studies have been done to improve the running time while maintaining the influence spread comparable (i.e., allowing a small loss) to that of SimpleGreedy [2–5,7,9,11–14,16,21,29,31]. In order to address the micro-level issue, the path based influence maximization (PB-IM) estimates the influence spread of a node by adding the weights of the paths from the node to other reachable nodes from it. Instead of performing costly MC-simulations, PB-IM requires a simple traversal of paths, which leads to a great amount of speed-up, thereby successfully remedying the micro-level issue.

One excellent approach tackling the macro-level issue [29,31] exploits the property of the *community structure* in a social network. They claim the influence spread within a single community is not that different from that in the whole social network. By exploiting this property, after choosing a new seed in a step, it re-evaluates the influence spread for *only those nodes in the community* where the seed has been selected. As a result, it significantly reduces the number of nodes requiring re-evaluation, thereby significantly improving the running time of SimpleGreedy. We refer to this approach as *community based influence maximization (CB-IM)*.

We found PB-IM and CB-IM, however, still have performance issues to be further elaborated. PB-IM resolves the micro-level issue but suffers from the macro-level issue. On the other hand, CB-IM resolves the macro-level issue but still faces the micro-level issue. This implies that each of PB-IM and CB-IM solves only one of the two *orthogonal* issues in micro and macro levels, leaving the other unsolved yet. In this paper, we propose Hybrid-IM, a hybrid method that combines PB-IM and CB-IM together for addressing both of the two issues in micro and macro levels.

Our Hybrid-IM consists of two stages. (1) Community detection: it partitions a whole social network into a number of communities in order to perform efficient re-evaluation of influence spread for nodes. (2) Seed selection: it performs k times to select and to add the most influential node into the k-seed set. It employs path based evaluation in computing the influence spread of a node and exploits the community structure detected in the previous stage in order to identify the nodes for re-evaluation.

Furthermore, we identify two more technical issues that could improve both accuracy and performance of Hybrid-IM. Accordingly, we propose two effective strategies to address them. The first issue is related to community detection in CB-IM. CB-IM builds a graph *over-simplified* from the original one by removing a very large number of edges in order to detect communities *fast*. While it is definitely beneficial in terms of the performance, it adversely affects the quality of the community structure due to a considerable loss of information. To address this issue, we propose a strategy called *path based community detection (PB-CD)* that enables to preserve most of the information in the original graph. By employing a *fast path based approach* to estimate the influence spread among communities, PB-CD could improve the quality of the community structure significantly, while performing community detection in a fairly short time.

The second issue is related to the cost-effective lazy forward (CELF) algorithm in CB-IM. The *CELF* reduces the number of re-evaluations by exploiting the *submodularity* of the influence function in IM, thereby effectively resolving the macro issue previously described [21]. This algorithm has been used in a number of its following approaches [4,5,12,16,31]. In CB-IM, a queue (hereafter referred to as a *local queue*) is assigned to each community and CELF is applied to all the queues *independently*. We point out some room (described in Section 3) here to be improved and propose *Global CELF (G-CELF)*, which significantly reduces the number of re-evaluations by effective elimination of unnecessary re-evaluations, while guaranteeing to produce exactly the same result as that by original CELF in CB-IM.

To show the effectiveness of our Hybrid-IM and the two strategies, we design the evaluation framework carefully and perform extensive experiments with four real-world datasets. When we detect the community structure of a social network with our PB-CD and select the k-seed set by using the communities, its influence spread becomes 28% larger than that obtained by original CB-IM. It indicates that our PB-CD detects communities more accurately than the original strategy in CB-IM. Moreover, we observe that our G-CELF improves the performance of original CELF in CB-IM by 30% in the seed selection stage. These two results indicate that our two strategies are effective in improving Hybrid-IM with respect to both accuracy and performance. Also, Hybrid-IM equipped with the two strategies is shown not only to improve the performance 100 times but also to select a good k-seed set whose influence spread is 28% larger, in comparison with CB-IM. Also, it outperforms PB-IM by 43 times in performance, preserving the influence spread of a seed set close (96.2% on average) to that obtained by PB-IM.

The paper is organized as follows. Section 2 briefly reviews previous work related to IM. Section 3 presents our Hybrid-IM and the two strategies for its refinement. Section 4 presents and analyzes experimental results that compare our Hybrid-IM and other state-of-the-art methods. Finally, Section 6 summarizes and concludes this paper.

2. Related work

2.1. Influence maximization

Influence Maximization (IM) is a problem of selecting a set S of k users whose influence spread over a social network is highest [6,27,36] among any combinations of k users. A social network is modeled by a graph defined as $G(V, E)$ where V

represents a set of nodes and E does a set of edges, each of which indicates a relationship between a pair of nodes having a weight showing its strength. Without loss of generality, this paper assumes a weight to have a range over (0,1]. The IM problem is formally defined as follows:

Influence maximization: Given $G(V, E)$ and budget k, it is defined as finding a set S consisting of k nodes, called a *k-seed set*, that maximizes the influence spread $\sigma(\cdot)$ of S over G.

$$S = argmax_{S \subset V, |S|=k} \sigma(S)$$

In computing the influence spread of a node, we need to define a model that explains how influence is propagated (i.e., spread) over the network [15,18–20]. The *independent cascade* (IC) model and the *linear threshold* (LT) model are widely accepted propagation models that capture the process of influence propagation in a discrete manner [15,25].

Finding the optimal solution to IM, however, is NP-hard because we need to consider *all possible* ${}_nC_k$ *k-seed sets*. Kempe et al. [15] proposed *SimpleGreedy*, a greedy method that adds a node having the maximum marginal gain over the network into the seed set at each step and repeats this step k times. The *marginal gain* of a node represents the influence spread *additionally* obtained by adding the node into the seed set. The marginal gain of node v is computed by $\sigma(S + \{v\}) - \sigma(S)$ where the influence spread function, $\sigma(\cdot)$, is estimated by running the MC-simulation, a method to get the final result by averaging the results of 10,000 simulations. In order to estimate $\sigma(S)$, IM by SimpleGreedy makes the nodes in S become *active* and starts the influence propagation process from S by the MC-simulations according to the given propagation model. When the process is finished, *the number of active nodes* in the network is regarded as the influence spread of S.

We know that a greedy algorithm guarantees to find an *approximate solution* that provides 63% quality of the optimal solution when the objective function satisfies all of the three conditions: (1) non-negativity, (2) monotonicity, and (3) submodularity. Kempe et al. [15] proved $\sigma(\cdot)$ for IM to satisfy all the three conditions. It is also noted that the influence spread obtained with SimpleGreedy is often considered as a *ground truth* in the IM problem.

SimpleGreedy, however, shows undesirable performance due to the problems occurring in two different levels: (1) at a micro level, it evaluates the influence spread of a node by *running 10,000 MC-simulations* for its stability, which is very time-consuming; (2) at a macro level, after selecting a new seed in a step, it re-computes the marginal gain of *all the non-seed nodes* for the next step because their marginal gain is likely to have been changed due to the selection of the new seed.

2.2. Existing solutions for IM

A number of research efforts have been devoted to address the two performance issues [2–5,7,9,11–14,16,17,21,29,31]. The *path based IM* (PB-IM) is a very promising solution that tackles the issue in a micro level [4,5,12,16]. It computes the influence spread of a node by summing the weights of all the paths to other nodes starting from itself, rather than running expensive MC-simulations. This method computes the influence spread of a node in a single traversal of paths, which is much faster than MC-simulations. Formally, a path starting from node v to node u is defined by $p = (v_1 = v, v_2, \ldots, v_m = u)(m \geq 2)$ where v_i denotes a node on the path from v to u. The weight of a path denoted as $W(p)$ is computed by

$$W(p) = \prod_{i=1}^{m} w(v_i, v_{i+1}), \tag{1}$$

where $w(v_i, v_{i+1})$ denotes the weight on the edge connecting v_i and v_{i+1}. The influence of node v is computed by

$$\sigma(v) = 1 + \sum_{u \in O_v} \sigma^u(v), \tag{2}$$

where $\sigma^u(v)$ and O_v denote the influence from node v to node u and a set of nodes reachable from node v. $\sigma^u(v)$ is computed by

$$\sigma^u(v) = 1 - \prod_{p \in P_{v \to u}} (1 - W(p)) \tag{3}$$

Finding all possible paths from a node, however, is known #P-hard [30]. PB-IM solves this problem by handling only those paths whose weights are larger than a pre-defined threshold. Ignoring some paths having quite small weights is unlikely to affect the accuracy of the total influence spread because the influence spread of a node mostly happens to its nearby nodes, i.e., located closely to it. As a result, PB-IM achieves significant performance improvement over SimpleGreedy up to three orders of magnitude while keeping its influence spread of the seed set almost identical (i.e., more than 99%) to that of SimpleGreedy.

To resolve the macro-level issue, Leskovec et al. [21] proposed cost-effective lazy forward (CELF) optimization that takes advantage of the *submodularity* of $\sigma(\cdot)$. The *submodularity* is that the marginal gain of a node reduces as the size of S increases. The objective function, $\sigma(\cdot)$, is *submodular* because it satisfies Eq. (4):

$$\sigma(S + \{v\}) - \sigma(S) \geq \sigma(T + \{v\}) - \sigma(T), \quad S \subset T \tag{4}$$

In other words, if marginal gain of node v computed in the *previous* step is smaller than that of another node u computed in the *current* step, marginal gain of node v in the *current* step should be smaller than that of node u. In this case, it is unnecessary to compute marginal gain of node v in the current step for selecting a seed. In this manner, CELF could reduce

a large number of unnecessary computations, thereby improving the performance of SimpleGreedy by up to 700 times while guaranteeing to provide exactly the same seed set as that by SimpleGreedy.

CB-IM also resolves the macro-level issue by exploiting the property of influence spread among communities of a social network [2,13,29,31]. The influence from a node tends to be spread *primarily* over the nodes in its belonging community and *rarely* over the nodes in other communities [8,35]. By exploiting this property, once a seed is selected in community C_i, CB-IM re-evaluates the marginal gain for only those nodes within C_i (i.e., rather than for all the nodes in the network), which reduces the number of re-evaluated nodes from n to n/m where n is the number of nodes in a network and m is the number of communities identified. CB-IM is known to provide about 15 times speed up with preserving 90% accuracy, compared with SimpleGreedy.

Arora et al. [1] performed a wide set of experiments that compare the influence spread, running time, and memory usage of existing IM algorithms and find such an IM algorithm that is preferable under different conditions. Yadav et al. [34] applied the influence maximization to real-world applications and addressed some challenges faced in real-world applications.

3. Proposed method

3.1. Overview

As explained in Section 1, each of PB-IM and CB-IM solves only one among the two issues in micro and macro levels, leaving the other unsolved yet. In this paper, we propose a hybrid approach called Hybrid-IM that combines PB-IM and CB-IM for addressing both of the two orthogonal issues together.

The simple combination of these CB-IM and PB-IM can further improve the running time but increases the loss of the influence spread of a seed set. Since CB-IM evaluates the influence spread of a node only within the community to which the node belongs, it cannot accurately evaluate the real influence spread of the node in the whole network, which may result in the loss of the influence spread of the selected seed set. PB-IM evaluates the influence spread of a node by ignoring the paths with weights below the pre-defined threshold because those paths have little effect on the node's influence spread and finding all the paths starting from a node is a #P-hard problem [30]. As the threshold gets smaller, the influence spread of nodes is more accurately evaluated while the computation time becomes larger. On the other hand, as the threshold gets larger, the computation time becomes smaller while the loss of the influence spread is increased more.

To reduce the loss of influence spread, we propose a novel community detection strategy (PB-CD). Moreover, we also propose the G-CELF strategy that improves the running time more, without any loss of influence spread. Hybrid-IM consists of two stages: community detection and seed selection.

Community detection. This stage divides the whole network into a number of communities for efficient seed selection in the next stage. It is also composed of the two sub-parts, *unit-community detection* (UCD) and *community merge* (CM).

Seed selection. Given a network divided into a set of communities in the previous stage, this stage selects a k-seed set. Hybrid-IM estimates the marginal gain of a single node by taking path based influence evaluation. Furthermore, after a seed is selected in step (i), Hybrid-IM reduces the number of nodes to be re-evaluated in step $(i+1)$ by exploiting the property of communities in a social network.

As a result, our Hybrid-IM addresses successfully the two issues in micro and macro levels. Furthermore, we refine Hybrid-IM in order to further improve its performance and reduce the loss of the influence spread of a seed set by proposing two strategies, PB-CD and G-CELF. In Sections 3.2 and3.3, we will discuss these two strategies in detail.

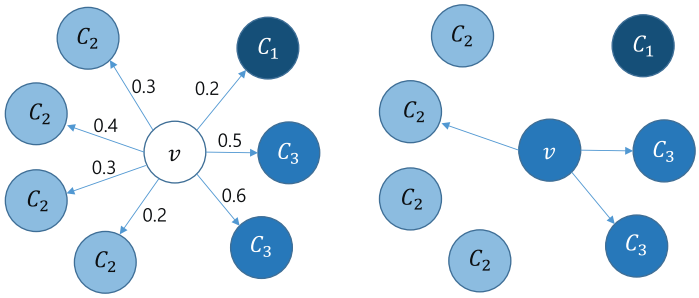
3.2. Path based community detection

In general, the quality of a community structure identified in a social network is considered high or good if it has a property that the nodes in the same community are tightly connected to each other (i.e., high intra-connectivity), while the nodes in different communities are loosely connected to each other (i.e., low inter-connectivity) [8]. In other words, the existing definition of the good community structure considers *only the connections* among nodes in a network. Because this definition totally ignores the *influence spread* of nodes among communities (i.e., the core notion in IM), it is inappropriate to be applied to the IM problem directly.

To this end, we need to *redefine the good community structure in our IM context*. In the new definition, the quality of a community structure identified in a social network is considered high or good if it has a property that *the influence of a node in a community is spread mostly over the nodes in the same community (i.e., high intra-spread), and is rarely spread over the nodes in different communities (i.e., low inter-spread)*. We thus define the community structure to be of high quality if the influence spread of nodes within a community is very close to that within the whole network. We note that this new definition coincides with the philosophy behind the idea in [29,31].

However, if communities are detected in such a way that the influence of a node is overflowed considerably to other communities, the difference between the influence spread of a node within a community and that on the whole network will be significant. This subsequently leads to finding a k-seed set of low quality (i.e., having smaller influence spread). Therefore, accurate community detection is quite important to find a k-seed set of high quality.

One of good ways to estimate accurately the influence spread between communities is to employ MC-simulations. Running MC-simulations, however, is very costly, which leads to a considerable amount of time required in community



(a) Original network. (b) Simplified network.

Fig. 1. Unit-community detection.

detection, thereby degrading the overall performance of CB-IM. For this reason, CGA [31], one of the CB-IM approaches, simplifies the original graph by removing a large number of edges, except for those (*live edges*) whose weight is larger than a pre-defined threshold, and considers the influence spread between communities only based on the live edges. Thus, the live-edge based method, employed in CGA [31], provides an improvement only in terms of running time.

While the small graph (over-) simplified from the original graph provides better performance, it causes significant information loss in community detection, incurring the problems as follows: (1) a large number of actual edges are ignored; we observed that 93% of the edges (on average) were removed in community detection of CB-IM when we follow the guideline in [31]; (2) the weights of live edges are ignored in the simplified graph; all live edges are treated identically. These two problems make significant influence overflowed via these ignored edges not captured in community detection, which leads to a failure of obtaining high-quality communities with respect to the influence spread.

In order to overcome these problems, we propose a new strategy of *path based community detection* (PB-CD) that relies on much more edges of the original graph (i.e., rather than only live edges) in estimating influence spread between communities. PB-CD exploiting richer information estimates the influence propagation more accurately, thereby finding such communities much more desirable for seed selection. Moreover, PB-CD performs the community detection task very efficiently because it takes path based evaluation of influence, rather than costly MC-simulations, unlike CB-IM. Therefore, it could achieve both goals of accuracy and performance. Our PB-CD consists of two steps: *unit-community detection* (UCD) and *community merge* (CM).

Unit-Community Detection. We assign a community label to each node based on its *affinity* to each neighboring community. The affinity of node v to a community C_m is defined by Eq. (5).

$$A(v, C_m) = \sum_{u \in (N(v) \cap C_m)} w(v, u) \tag{5}$$

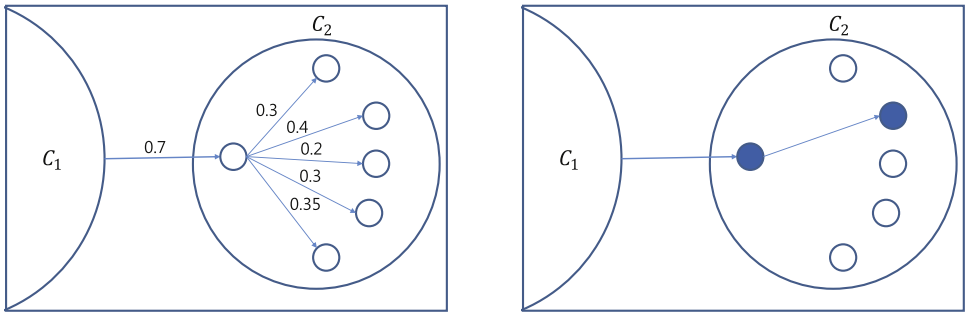
where $w(v, u)$ and $N(v)$ indicate a weight of an edge from v to u and the set of neighboring nodes of v, respectively. The unit-community detection runs as follows:

(1) Initially, we assign a unique community label (i.e., node id) to every node v.
(2) For each node v,
  (1) We compute its affinity to each of its directly connected community.
  (2) We assign v with the label of the community having the maximum $A(v, C_m)$.
(3) We repeat (2) until the labels of nodes do not change any more.

Raghavanet al. [26] showed that the structure of unit-communities becomes *stable* if this process is repeated by a sufficient number of times. Following [26], we performed the unit-community detection step repeatedly, making it stop when the community labels of all the nodes no longer change. In other words, it terminates when the affinity of every node to its own community is the maximum.

Fig. 1 shows how node v is assigned to a community in UCD. Fig. 1(a) shows the original network composed of three communities (C_1, C_2, C_3) where v is not assigned to any community yet. Fig. 1(b) shows the network simplified by only those live edges having a weight larger than the threshold of 0.4. CB-IM treats live edges identically regardless of actual edge weights. So, it assigns C_3 to v as in Fig. 1(b) because v has two adjacent C_3 nodes and one adjacent C_2 node. On the other hand, PB-CD assigns C_2 to v because the *affinity* of v to C_2, $A(v, C_2) = 0.3 + 0.4 + 0.3 + 0.2 = 1.2$, is larger than that of v to C_3, $A(v, C_3) = 0.5 + 0.6 = 1.1$. Unlike CB-IM, PB-CD uses not only non-live edges but also their weights to find more intimate communities in UCD.

Path Based Community Merge. The community merge (CM) step is to merge a pair of communities by considering the overflowed influence between the two communities after the unit-community detection (UCD) step. If the amount of the overflowed influence between two communities is larger than a pre-defined threshold, it merges the two communities. This merge is necessary because these two communities do not satisfy the property of a good community structure in the IM perspective, which requires the influence spread of a node in a single community not to be so different from that in the



(a) Original network. (b) Simplified network.

Fig. 2. Process of community merge.

whole network. On the other hand, if the amount of overflowed influence between two communities is smaller than the threshold, it indicates that they are separated well and satisfy the property of the community structure. Therefore, the two communities should remain separately without merging.

The overflowed influence from C_m to C_l is defined by Eq. (6) below. This is the maximum one among the overflowed influence by every possible pair of nodes, one in C_m and the other in C_l. It is highly likely that the maximum overflowed influence is determined by the node pair where the seed candidates with high influence spread are involved. In other words, communities are detected from the network, considering the seed nodes that have high influence spread over the network.

$$IOF(C_m, C_l) = \max_{v \in C_m, u \in (N(v) \cap C_l)} \frac{w(v,u) \cdot \bar{\sigma}_m(\{v\})}{\sigma_m(\{v\})} \tag{6}$$

$\sigma_m(\{v\})$ and $\bar{\sigma}_m(\{v\})$ in Eq. (6) indicate the influence of node v within C_m and the influence of u outside of C_m, respectively. Thus, as $IOF(C_m, C_l)$ is larger, the more influence is overflowed from C_m to C_l. The process of community merge performs as follows:

(1) For any pair of communities C_m and C_l,
  (1) We compute $IOF(C_m, C_l)$.
  (2) If $IOF(C_m, C_l)$ is larger than the pre-defined threshold, we merge C_m and C_l.
(2) We repeat (1) until no pairs have $IOF(C_m, C_l)$ smaller than the pre-defined threshold.

Fig. 2 shows an example that the overflowed influence is *under-estimated* with the existing method in CB-IM. Fig. 2(a) shows the original network composed of two communities (C_1, C_2). Fig. 2(b) shows the network simplified by using only live edges having a weight larger than the threshold of 0.4 (i.e., the threshold 0.4). The existing method estimates the amount of influence overflowed between communities by considering only the live edges, thereby ignoring the amount of propagation through a much more number of non-live edges. This leads to the inaccurate estimation of the amount overflowed influence between communities.

Our PB-CM, however, estimates an *affinity* of node to a community and the influence overflowed between communities by considering not only both of live and non-live edges but also their weights, which leads to more accurate estimation. Therefore, our PB-CD strategy could help identify communities of higher quality for more accurate seed selection.

When there is a user who has great influence spread in the network, her/his influence could spread beyond the community which she/he belongs to. It is an inevitable loss when we take advantage of the property of the community structure to improve the performance. However, our community detection method detects communities from the network in such a way that it guarantees the influence of a user that spreads to other communities does not exceed the pre-defined value. Therefore, this loss is not that much. Algorithm 1 is the whole process of our PB-CD.

3.3. Global-CELF

CELF resolves the macro issue of SimpleGreedy by exploiting the *submodularity* of the marginal gain function. In CB-IM, a queue (referred to as a *local queue*) is assigned to each community where nodes are sorted in descending order of their marginal gains. CELF is applied to communities' local queues *independently*. The whole process of original CB-IM proceeds as follows:

(1) We evaluate the marginal gain of every node by running MC-simulations.
(2) We select the node having the maximum marginal gain in each community as a *seed candidate*.
(3) Among the seed candidates from all communities, we finally select the seed candidate with the maximum marginal gain as the seed at the current step.
(4) We re-evaluate the marginal gains of all the nodes belonging to the community where the above seed has been selected.

Algorithm 1 Path Based Community Detection (PB-CD).

Input: network $G(V, E)$, unit-community threshold τ, merge threshold θ
Output: a set of communities $C = \{C_1, C_2, \ldots, C_M\}$

```
for each v ∈ V do
    v.C ← a unique community label;
end for
for t = 1 to τ do
    for each v ∈ V do
        maxAffinity ← 0;
        for each C_m ∈ NeigborCommunity(v) do
            if A(v, C_m) > maxAffinity then
                maxAffinity ← A(v, C_m); v.C ← C_m;
            end if
        end for
    end for
end for
isExist ← True;
while isExist do
    isExist ← False;
    for each C_m ∈ C do
        for each C_l ∈ C − C_m do
            if IOF(C_l, C_m) > θ then
                C_l ← C_m; isExist ← True;
                break;
            end if
        end for
    end for
end while
return C;
```

(5) We select the node having the maximum (re-evaluated) marginal gain in this community as the seed candidate for the community.
(6) We repeat (2)–(5) until the seed set size becomes k.

Fig. 3(a) shows the process of CELF in CB-IM where three communities C_i, C_j and C_k exist. Each bar represents a local queue and each box in a bar does a node where the number is the marginal gain of the node. A colored box indicates the node that has already been re-evaluated and an uncolored box indicates the node that has not been re-evaluated yet. For selecting the first seed, every box is colored because the marginal gain of every node should be evaluated. The seed candidate, node a, in C_i is selected and popped out as the seed because it has the maximum marginal gain (24) at step (1). CELF is applied to community C_i where the new seed (i.e., node a) has been selected. The marginal gain re-evaluation starts from node b, continues with its following nodes (i.e., c and d) in the local queue one by one, and stops with node e because its marginal gain (12) (not evaluated yet at step 2) is smaller than or equal to the marginal gain (12) of the above node (i.e., c). During this re-evaluation, node d is selected as a new candidate for C_i for step 2 as in Fig. 3(a). In seed selection, node f in C_j having the maximum marginal gain is selected among seed candidates as the next seed at step 2. This process is repeated until the size of the seed set becomes k.

In CB-IM, after a seed is selected, CELF is *independently* applied to a local queue of the community to which the seed belongs, without any consideration on the marginal gains of seed candidates from other communities.

Our Lemma 3.1 shows a potential that we can further eliminate unnecessary re-evaluations of marginal gains. Before stating Lemma 3.1, we explain a property of the marginal gain of a node in CB-IM. If a seed has *not* been selected from $C_j (i \neq j)$ during step l to step m, the marginal gain of the seed candidate sc_j from C_j is not changed as shown in Eq. (7):

$$\sigma(S^m + \{sc_j\}) - \sigma(S^m) = \sigma(S^l + \{sc_j\}) - \sigma(S^l) \quad (m > l) \tag{7}$$

This indicates that the seed selected from C_i does not change the marginal gain of nodes in C_j in CB-IM. This is because the influence spread of a node is only considered *within* the community.

Lemma 3.1. *Given a graph $G(V, E)$, let sc_j be a seed candidate in C_j, and S^m and S^l be the seed set at step m and step l, respectively where $m > l$. For any $u \in C_i$, $(i \neq j)$, suppose that the marginal gain of sc_j at step m is greater than the marginal gain of u at step l, i.e.,*

$$\sigma(S^m + \{sc_j\}) - \sigma(S^m) \geq \sigma(S^l + \{u\}) - \sigma(S^l)$$

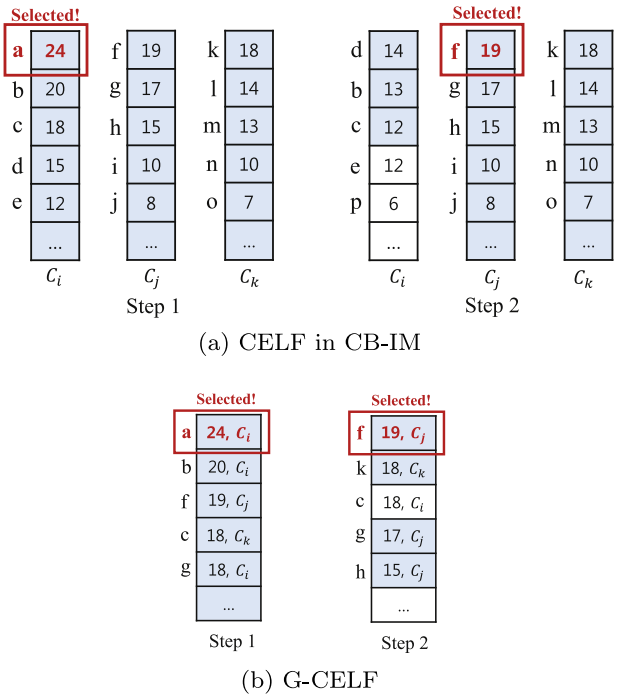


Fig. 3. Comparison of CELF in CB-IM and G-CELF.

Then, the marginal gain of u at step m cannot be greater than that of sc_j at step m, i.e.,

$$\sigma(S^m + \{sc_j\}) - \sigma(S^m) \geq \sigma(S^m + \{u\}) - \sigma(S^m)$$

Proof for Lemma 3.1. By *submodularity*, it can be observed that

$$\sigma(S^l + \{u\}) - \sigma(S^l) \geq \sigma(S^m + \{u\}) - \sigma(S^m) \quad (m > l)$$

$$(\because S^l \subset S^m)$$

Then, it follows that

$$\begin{aligned}\sigma(S^m + \{sc_j\}) - \sigma(S^m) &\geq \sigma(S^l + \{u\}) - \sigma(S^l) \\ &\geq \sigma(S^m + \{u\}) - \sigma(S^m)\end{aligned}$$

$$\therefore \sigma(S^m + \{sc_j\}) - \sigma(S^m) \geq \sigma(S^m + \{u\}) - \sigma(S^m)$$

By Lemma 3.1, we could exploit the marginal gains of the seed candidates of the other communities to reduce the number of nodes that need to be re-evaluated in a local queue corresponding to a current community. We note some of the nodes in the local queue could never become the next seed node in spite of the re-evaluation of their marginal gains, if (at least) one of the other communities has a seed candidate whose marginal gain is larger than their *current* marginal gain. Therefore, in the case of re-evaluating the marginal gains of the nodes in the local queue for the next seed selection, for any node v, if a seed candidate in other communities satisfies Lemma 3.1, the marginal gain of v does not need to be re-evaluated since it cannot become the seed in the next step. Corollary 3.2 specifies the condition of nodes that cannot be the seed in the next step. □

Corollary 3.2. *For any sc_j and $u \in C_i (i \neq j)$, if at least one sc_j exists satisfying $\sigma(S^m + \{sc_j\}) - \sigma(S^m) \geq \sigma(S^l + \{u\}) - \sigma(S^l)$, u cannot be the seed at the current step.*

Proof for Corollary 3.2. By Lemma 3.1, the marginal gain of u at step m cannot be greater than that of sc_j. Thus, u cannot be the seed at the step.

By exploiting Corollary 3.2, we can reduce the running time of the seed selection process by skipping the unnecessary re-evaluations of marginal gains for the nodes satisfying the following conditions. If the node from C_i is selected and popped out as a seed at step l, the marginal gains of nodes in other communities $C_j (i \neq j)$ are not changed (Eq. (7)) for the seed selection at step m, and the re-evaluation process only for the nodes within C_i starts in CB-IM. However, if there exists at

least one sc_j in other community $C_j (i \neq j)$ that has greater marginal gain at step $m(>l)$ than node u in C_i, the re-evaluation process of all nodes below u in C_i is unnecessary by Corollary 3.2.

Fig. 3(a) shows the inefficiency of CELF in CB-IM where Corollary 3.2 is not applied: node a having the largest marginal gain has been selected among seed candidates as the seed in step 1. The re-evaluation process starts from node b. The re-evaluation of node b is necessary because the marginal gain (20) of node b is greater than that (19) of the seed candidate (node f) in C_j. Because the marginal gain (18) of node c, however, is smaller than that (19) of node f, its re-evaluation is *not necessary* by Corollary 3.2. However, the marginal gain of node c is re-evaluated in CB-IM because CELF is *independently applied* to local queues *without considering the marginal gain of the nodes in other communities*.

In addition, CB-IM compares the marginal gains of all the seed candidates to select a current seed at each step. When the number of communities is M, the time complexity of this comparison is $O(M)$, which can be reduced to $O(log(M))$ if we employ other data structures such as a *winner tree*. Considering that M easily becomes several tens, this is not negligible in terms of performance.

We propose a *Global CELF* (G-CELF) strategy that (1) eliminates unnecessary re-evaluations of marginal gain by exploiting Corollary 3.2 and (2) does not require to compare the marginal gains of seed candidates at each step. In contrast to CELF in CB-IM, G-CELF *maintains only a single queue*, called a *global queue*, whose entry keeps not only the unique id (*id*) and marginal gain (*mg*) but also the community label (*comm*) and a flag (*u.flag*) indicating the step number when the node was re-evaluated most recently. Additionally, each community has the step number (*C.flag*) when a seed was selected most recently from the community. Each entry is sorted in descending order of the node's marginal gain in a single global queue. The community label and flag information are necessary to identify the community of a node in a single global queue and to decide whether to re-evaluate the top node or not.

The whole process of G-CELF is described in Algorithm 2. The marginal gains of all nodes are evaluated for the first seed

Algorithm 2 G-CELF.

Input: network $G(V, E)$, seed size k, number of communities M
Output: a seed set S

```
1: Q ← ∅; S ← ∅;
2: C_1.flag = C_2.flag = ··· = C_M.flag = 0;
3: for each u ∈ V do
4:    u.mg = σ({u}); u.flag = 1;
5:    Add u to Q;
6: end for
7: while |S| < k do
8:    u=top entry (node) in Q;
9:    if u.flag > C_u.comm.flag then
10:      S ← S ∪ {u}; Q ← Q − {u};
11:      C_u.comm.flag = |S|;
12:   else
13:      u.mg = σ(S ∪ {u}) − σ(S);
14:      u.flag = |S| + 1;
15:      Reinsert u into Q and re-order it according to u.mg;
16:   end if
17: end while
18: return S;
```

selection in step 1 and maintained in the queue (lines 3–6). The top node is selected as the seed at step 1 and popped out of the queue. Then, starting from the new top node, the process of re-evaluation starts inAlgorithm 3 order to select next seeds until the size of the seed set reaches k (lines 7–17). For the new top node, the re-evaluation process of G-CELF starts. If the new top node *has been re-evaluated* since the most recent seed from its belonging community was selected, the re-evaluation of the node is unnecessary and the node is selected as a seed at the current step (lines 9–11). If the new top node *has not been re-evaluated* since the most recent seed from its belonging community was selected, the node is re-evaluated and re-ordered in the queue according to its new marginal gain (lines 12–15). □

Lemma 3.3. *After the re-evaluation process of G-CELF is finished, the top node in the global queue always has the maximum marginal gain in the next step.*

Proof for Lemma 3.3. Since G-CELF re-evaluates the marginal gains of the nodes in the order from the top to the bottom in the global queue, it is natural that the marginal gains of nodes from other communities are considered together unlike CELF in CB-IM. Thus, Corollary 3.2 holds in G-CELF, which indicates that there are no unnecessary re-evaluations occurring in CB-IM. The nodes not re-evaluated in G-CELF cannot be a seed at the step (Corollary 3.2), thus, no re-evaluation of these nodes does not affect seed node selection. As a result, the top node selected as the current seed by G-CELF has the maximum marginal gain in the step and thus is identical to that selected by CELF in CB-IM.

Table 1
Dataset statistics.

Dataset	NetHEPT	NetPHY	Stanford	DBLP
# of nodes	15K	37K	281K	655K
# of edges	58K	231K	2.31M	3.98M
Max in-degree	7.7	12.4	8.2	6.1
Max out-degree	341	286	38,606	588
Direction	Undirected	Undirected	Directed	Undirected

This implies that our G-CELF selects the node with the maximum marginal gain exactly same as that by CELF in CB-IM (i.e., without any loss of the influence spread), even though the number of marginal gain re-evaluations of our G-CELF is smaller than that of CELF in CB-IM. If we repeat our G-CELF k times, we can obtain a k-seed set that is exactly the same as the k-seed set obtained by CELF in CB-IM (Corollary 3.4) □

Corollary 3.4. *Given a graph $G(V, E)$ and the size of the seed set k, our G-CELF strategy selects a k-seed set which is identical to the seed set by CELF in CB-IM.*

Corollary 3.5 (Proof for Corollary 3.4)**.** *We proved that the node selected as the current seed by G-CELF is identical to that by CELF in CB-IM in Lemma 3.3. Therefore, we can get a k-seed set that is identical to that obtained by CELF in CB-IM by iterating the same process k times.*

Fig. 3(b) shows an example case handled by G-CELF. The top node a in C_i is selected as the seed in step 1. The re-evaluation process starts from the next node b. Because node b has not been re-evaluated since a seed (node a) was selected from its belonging community (C_i), the marginal gain of node b is re-evaluated (line 14). Node f, however, has already been re-evaluated at step 1 and its marginal gain is not changed because node f belongs to a different community C_j. Thus, its re-evaluation is not required and node f is selected as the second seed (line 10). In this example, the number of marginal gain re-evaluations for selecting two seeds is only 1 in G-CELF while it is 3 in CELF in CB-IM. G-CELF performs faster than CELF in CB-IM by re-evaluating less nodes compared to CELF in CB-IM while guaranteeing to find the exact same seed set *(i.e., no loss of accuracy). Algorithm 3*

Algorithm 3 Hybrid-IM.

Input: network $G(V, E)$, seed size k, unit-community threshold τ, merge threshold θ
Output: a seed set S
1: $C \leftarrow PB - CD(G, \tau, \theta)$;
2: $M \leftarrow |C|$;
3: $S \leftarrow G - CELF(G, C, k, M)$;
4: return S;

is the whole process of our Hybrid-IM.

4. Evaluation

In this section, we evaluate the effectiveness of our Hybrid-IM with four real-world datasets by answering the following key questions in four categories:

(1) Parameter tunning
  (a) What is the best pruning threshold α for efficient path based influence evaluation on each dataset?
  (a) What is the best edge selection threshold β for community detection on each dataset?
(2) PB-CD Strategy
  (a) Does our PB-CD consider the influence among communities in community detection more accurately than the existing one in CB-IM?
  (b) How is the running time of our PB-CD, compared with the existing one employed in CB-IM?
(3) G-CELF Strategy
  (a) How much does our G-CELF improve the performance of CELF in CB-IM?
(4) Hybrid-IM
  (a) How much does our Hybrid-IM improve the performance of existing IM methods?
  (b) How accurate is our Hybrid-IM in terms of influence spread in comparison with the existing IM methods?

4.1. Experimental setup

Dataset. We used four real-world datasets, NetHEPT, NetPHY [4], DBLP [5], and Stanford [22] whose statistics are summarized in Table 1. NetHEPT, NetPHY [4] and DBLP [5] are the datasets of a co-authorship network. Stanford [22] is the dataset of a web graph consisting of web pages and their hyperlinks.

Table 2
Best values of α for four datasets.

Dataset	NetHEPT	NetPHY	Stanford	DBLP
Threshold	1/320	1/640	1/160	1/320

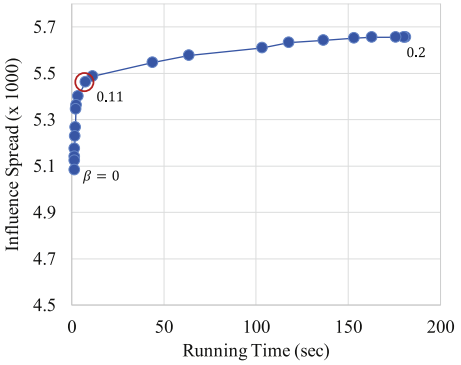


Fig. 4. Influence spread and running time with different β (NetHEPT).

Table 3
Best values of β for four datasets.

Dataset	NetHEPT	NetPHY	Stanford	DBLP
Threshold	0.11	0.09	0.2	0.2

Diffusion Model. All experiments were conducted under the weighted cascade (WC) model [15,28], which is a variation of the IC model. Following [10,15], we assigned the weight on edge (u, v) by $1/idegree(v)$ where $idegree(v)$ indicates the number of in-coming edges of node v.

4.2. Parameter tuning

Q1.1. Best threshold for path pruning: α

In this experiment, we try to find the best threshold α for path pruning. If the weight of a path is smaller than α, the path is not used for influence evaluation. This experiment aims to find the threshold value that optimizes both accurate influence estimation and fast running time. We recorded the running time and influence spread of a seed set at every α of $1/5, 1/10, 1/20, 1/40, \ldots, 1/1280$. The best values for Stanford and DBLP are already suggested in [16]; we performed this experiment only for NetHEPT and NetPHY. Table 2 shows the results providing the best values of α for four datasets. We used this set of values for path pruning in the following experiments.

Q1.2. Best threshold for edge selection in community detection: β

This experiment aims to analyze the distribution of edges in each network according to their weights for understanding those edges useful in community detection. We can reduce the running time of community detection significantly by excluding the edges that have negligible effect on the result of community detection.

The experiment was as follows: we first examined the distribution of edges in each dataset by increasing β from 0 to 1 in step of 0.01; we selected a 1,000-seed set by Hybrid-IM with each value for β; we recorded the influence spread of each 1,000-seed set and the running time required to obtain it. In all datasets, most of edges were shown to be in the range of weights less than 0.5. More specifically, the edges in the range of weights less than 0.2 are more than 50% of the total edges.

Fig. 4 shows the influence spread and running time of the 1,000-seed set selected with each value for β in NetHEPT. The x-axis represents the running time and the y-axis does the influence spread. In the range from 0.00 to 0.11 for β, the influence spread is rarely reduced (i.e., 4%) while the running time is significantly (i.e., 43 times) reduced. However, if β is larger than 0.2, the loss of influence spread starts to become larger than 10% although its running time is reduced less significantly than that with 0.11. In this regard, we chose 0.11 for the final value of β in NetHEPT, which provides significant performance improvement with insignificant accuracy reduction. We also set the best values of β for other datasets in the same manner as shown in Table 3.

Table 4
Community detection (CD) methods.

UCD \ CM	Live based	Path based
Live based	LL_CD	LP_CD
Path based	PL_CD	PP_CD

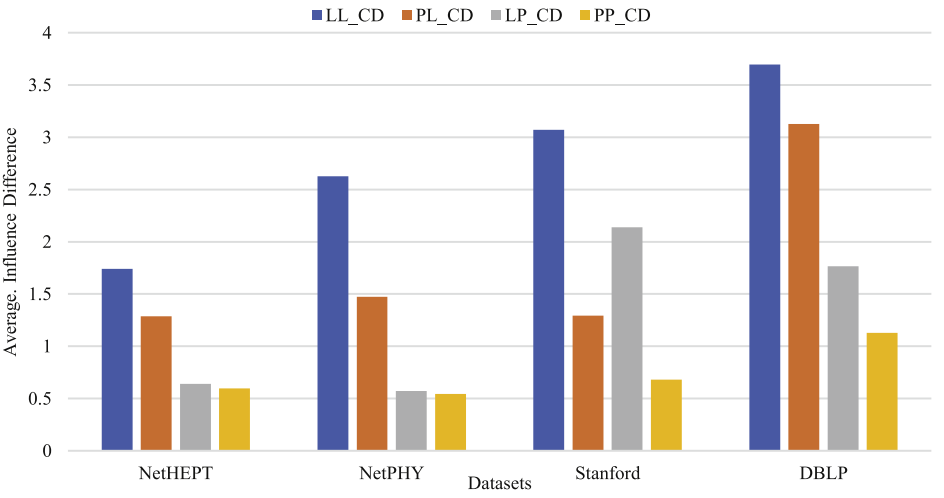


Fig. 5. Difference of influence spread in a community and an entire network.

4.3. PB-CD Strategy

We performed the experiment to verify the effectiveness of our PB-CD. First, to answer Q2.1, we examine whether PB-CD detects a better community structure by considering the influence overflowed between communities than existing methods. Second, to answer Q2.2, we compared the running times of community detection by our and existing methods.

Q2.1. Overflowed influence

We define a set of communities to be of high quality if the influence spread of a node within a community is very close to that within the whole network. In this paper, we claim that the influence spread of the seed set selected in a set of communities of high quality is larger than that of the seed set selected in a set of communities of low quality.

To demonstrate the effectiveness of our PB-CD in the IM context, we performed the following two experiments. After detecting communities using each community detection method, we first compared the influence spread of a node in the whole network with that within a community. We also examined the total influence spread of the k-seed set selected from the set of communities obtained by each of different community detection methods. These two experiments could show the effectiveness of the proposed PB-CD directly and indirectly over existing ones.

We set the best threshold for community merge in the same manner as used in [31]: we detect the community with increasing the threshold from 0.1 to 0.6 in step of 0.1 and finds the best one in terms of influence spread and performance.

In order to show the effectiveness of two sub-parts (i.e., unit-community detection and community merge) *individually*, we build *four* methods of community detection by all possible combinations, which are described in Table 4, where *rows* indicate how to detect unit-communities and *columns* do how to merge communities.

LL_CD refers to the community detection with live-edge based UCD and live-edge based CM, which is equal to CB-IM's original community detection method, while PL_CD refers to that with path based UCD and live-edge based CM, which is to show the effectiveness of the path based method in UCD. LP_CD refers to that with live-edge based UCD and path based CM, which is to demonstrate the effectiveness of the path based method in CM. Finally, PP_CD refers to the community detection method with path based UCD and path based CM, which is equal to our PB-CD.

Fig. 5 shows the average difference between the influence spread of a node in the whole network and that within a community. The proposed PB-CD (i.e., PP_CD) shows the smallest difference between the influence spread of a node in the whole network and that within a community among all the competing methods. This indicates that the PB-CD is the best method that detects the communities of high quality in the IM context.

Fig. 6 shows the results in terms of the influence spread. The x-axis represents CD methods, and the y-axis does the influence spread. We observe that PL_CD improves LL_CD by 2.2% on average, implying that path based UCD is more effective than the live-edge based one. This is because live-edge based UCD ignores a number of non-live edges whereas our path based UCD uses most of edges and even their weights.

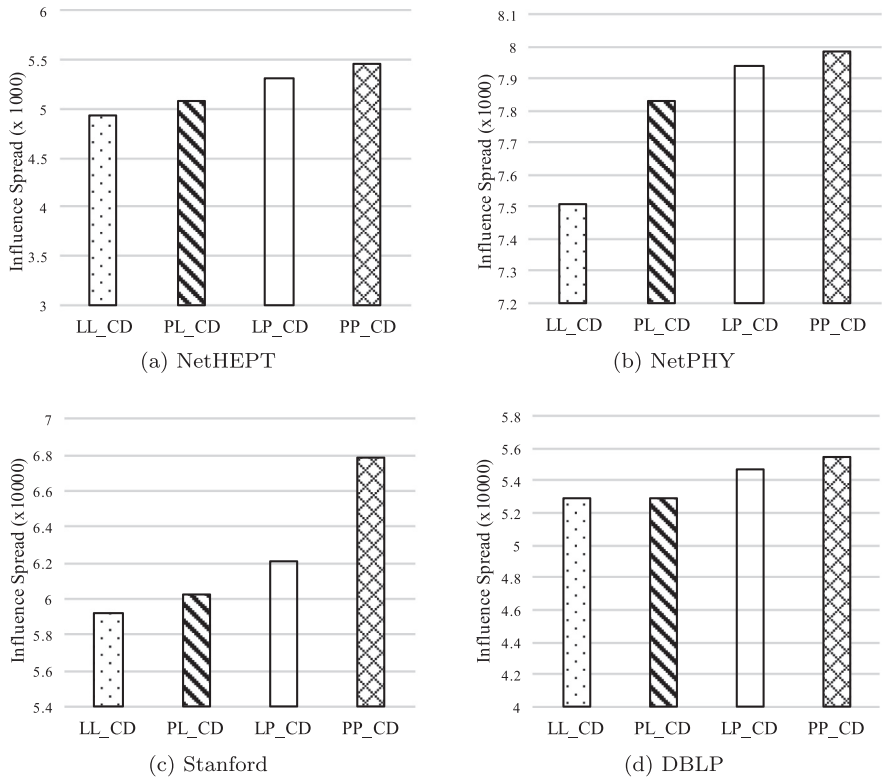


Fig. 6. Comparison of CD methods in terms of influence spread.

The results of LL_CD and LP_CD show that our path based CM provides larger influence spread than the live-edge based CM by 7.5%, 5.5%, 4.7%, and 3.3% in NetHEPT, NetPHY, Stanford, and DBLP, respectively. This is because the over-simplified graph obtained by live-edge based CM has a large information loss (i.e., edges and weights), which leads to the failure of performing correct community merge. On the other hand, our path based CM utilizes more information related to edges and weights, which could help find a good community structure that accurately considers the overflowed influence among communities.

Finally, our method, PP-CD, is shown to outperform universally *all* other combinations for *all* the datasets. The results of LL_CD and PP_CD show that the our PB-CD provides larger influence spread by 9.8%, 6.1%, 12.8%, and 4.6% in NetHEPT, NetPHY, Stanford, and DBLP, respectively, than CB-IM's original community detection. The result shows that the communities detected by our PB-CD help find a seed set providing the biggest influence spread. Based on the two results, we conclude that the proposed PB-CD is the most effective in detecting high-quality communities in the IM context.

Q2.2. Running time

This experiment is to show the degree of performance improvement by our path based strategy in community detection. We recorded the running time of the four combinations in Table 4. Fig. 7 shows the results with four datasets. The *x*-axis represents CD methods, and the *y*-axis does their running times.

PL_CD shows a result somewhat slower than or comparable to that by LL_CD. This is because PL_CD requires an additional overhead to compute the affinity score of every node and uses more edges than LL_CD. LP_CD is shown to perform worse than LL_CD with all four datasets because LP_CD takes more time in CM to process much more edges and their weights. However, since LP_CD uses path based influence evaluation, it performs community detection within a reasonable time. Note that the results of LP_CD in terms of influence spread are much superior to those of LL_CD.

Finally, our PB-CD (i.e., PP_CD) is a little slower than the existing method (i.e., LL_CD). This is because LL_CD has better performance owing to an over-simplified graph while PP_CD detects communities using much more edges and their weights in both UCD and CM. In Section 4.5, we will compare the total running times of Hybrid-IM and CB-IM in more detail.

4.4. G-CELF Strategy

Q3. Running Time

In this experiment, we compare the running times of two Hybrid-IMs to select 1000 seeds using our G-CELF and using original CELF in CB-IM. Fig. 8 shows the running times on four datasets. The *x*-axis represents the strategies equipped, and the *y*-axis does the total running times. We see that G-CELF consistently outperforms the existing one by 21.1%, 26.1%, 2.1%,

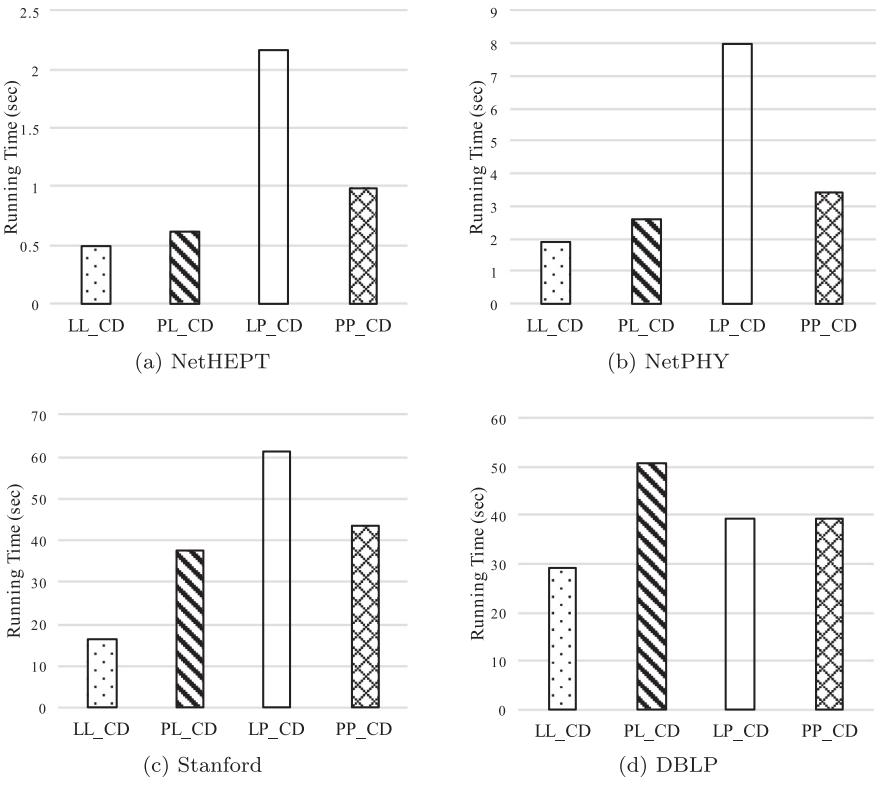


Fig. 7. Comparison of CD methods in terms of running time.

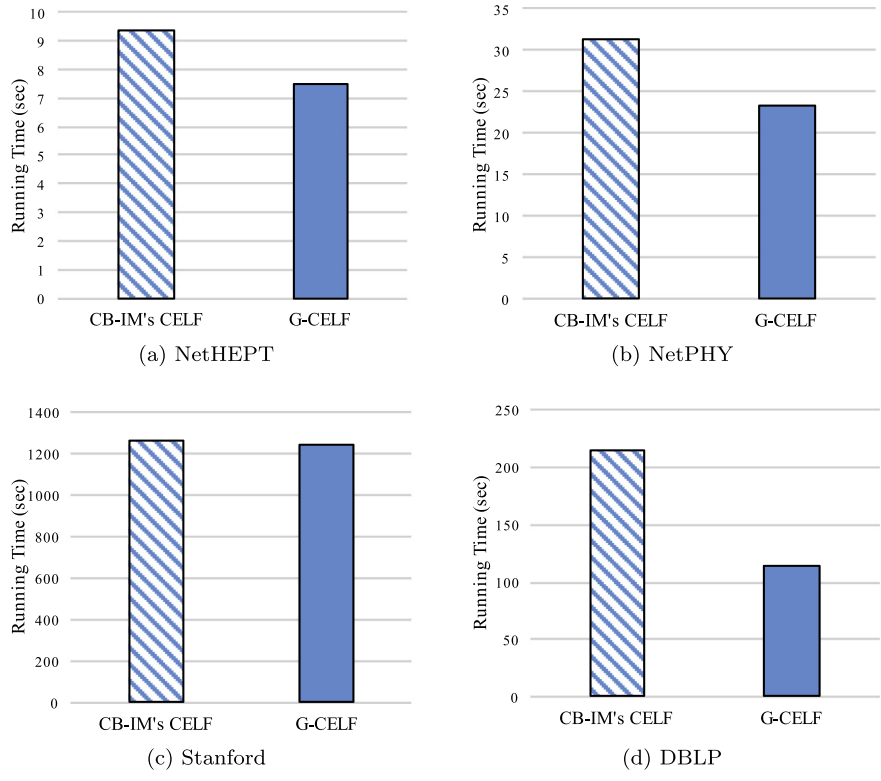


Fig. 8. Comparison of CELF strategies (running time).

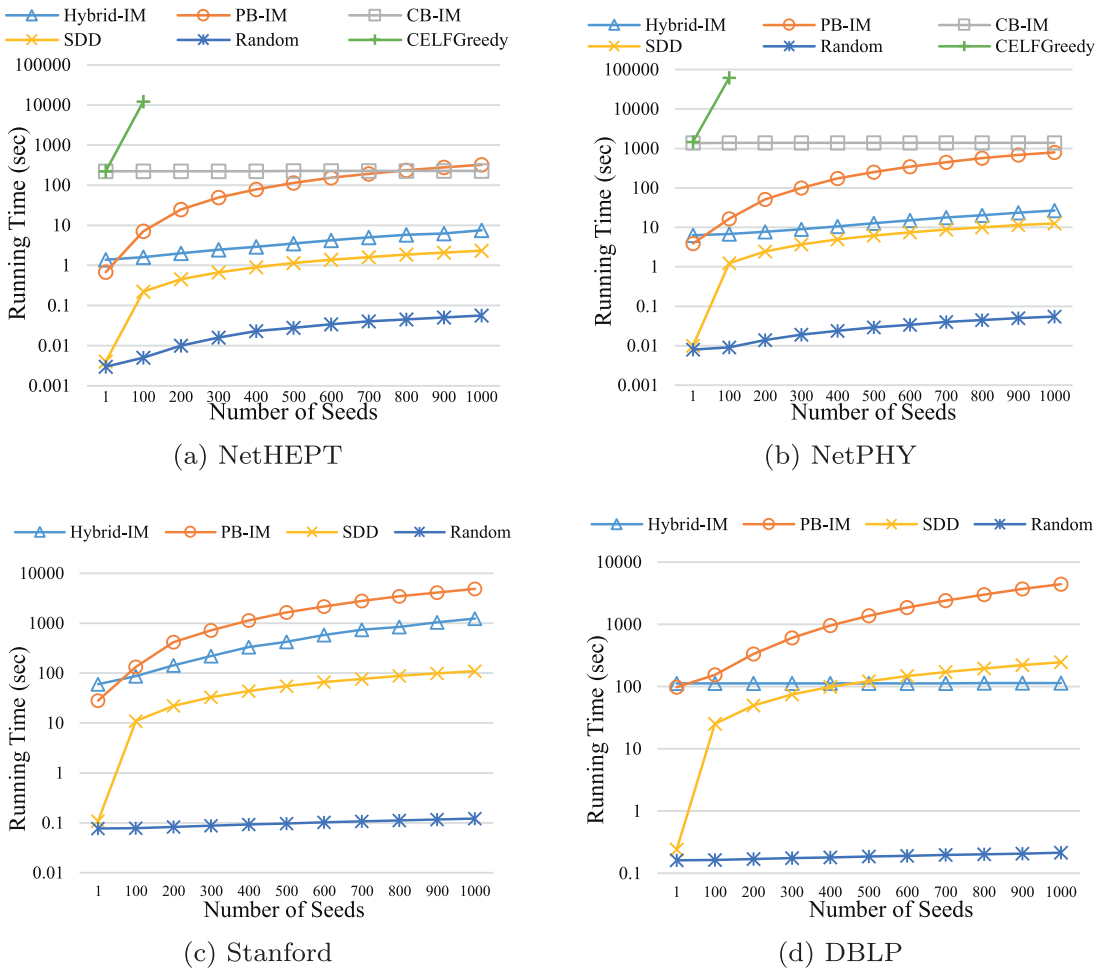


Fig. 9. Comparison of IM methods (running time).

47.1% in NetHEPT, NetPHY, Stanford, and DBLP, respectively. The reasons of the improvement are two folds: (1) the number of marginal gain re-evaluations for nodes is reduced a lot; (2) the comparisons among seed candidates from communities are not required any longer.

4.5. Hybrid-IM

Q4.1. Running Time

Let us now examine the effectiveness of our Hybrid-IM in comparison with other state-of-the-art methods. We measured the running time of each method with different sizes (1 to 1000) of a seed set.

Methods. We compare the following IM methods: *Random* is a baseline that selects a seed node randomly in each step. SDD (single degree discount) selects a node having the highest degree; after a seed is selected, the degree of all its neighbors is decreased by 1. *CELFGreedy* is the greedy algorithm with CELF [21]. It runs 10,000 MC-simulations to estimate the influence spread of a seed set. We chose CGA [31] as CB-IM, which is to select k-seed set after the live-edge based community detection (in the same wasy as LL_CD in Section 4.3) is performed, and chose IPA [16] as PB-IM, which performs in the same way as Hybrid-IM that does not employ the community detection stage. Finally, Hybrid-IM is our proposed one.

Fig. 9 shows the results where the x-axis represents the number of seeds and the y-axis represents their running time (log-scale). Note that we include the time spent to detect communities when measuring the running time of CB-IM and Hybrid-IM for fair comparisons.

Hybrid-IM outperforms all existing IM methods for every dataset, except for *Random*. *Random* is very fast but provides the seed set of extremely low quality. Following Hybrid-IM, PB-IM and SDD show relatively short running times. Finally, CB-IM performs worst. The execution of CB-IM and CELFGreedy is not completed in a reasonable time with Stanford and DBLP, and thus their results are not reported here in Figs. 9 and 10.

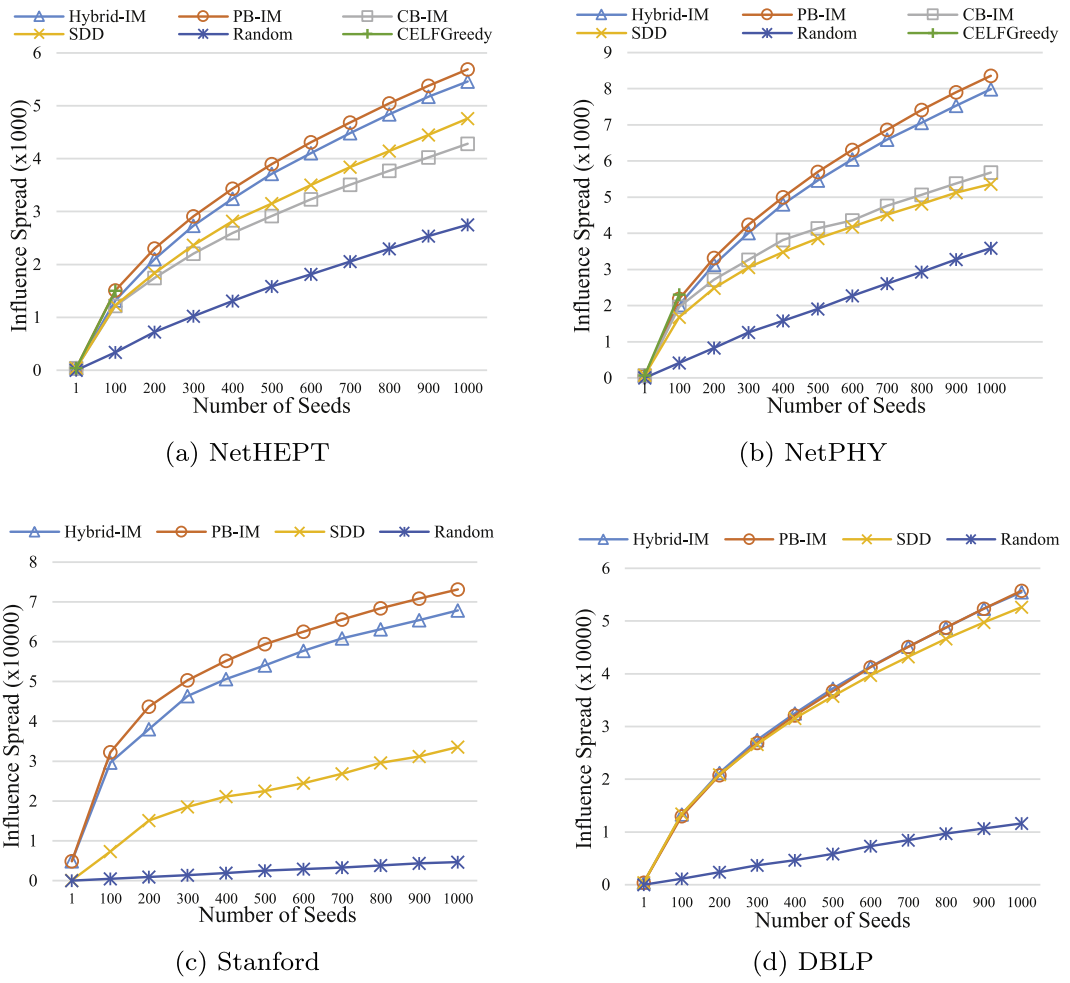


Fig. 10. Comparison of IM methods (influence spread).

Hybrid-IM selects a 1000 seed set about 100 times faster than CB-IM in both NetHEPT and NetPHY. This is because Hybrid-IM solves the micro issue by estimating the influence spread of a node without running MC-simulations. The running times of Hybrid-IM and CB-IM increase more slowly, compared with others, as the size of a seed set grows. This is because they successfully reduce the number of re-evaluations by taking advantage of the property of communities in a social network.

Also, Hybrid-IM outperforms PB-IM for all the datasets. Initially, Hybrid-IM shows similar performance to PB-IM. This is because Hybrid-IM requires a fixed amount of time for community detection. However, it starts to perform better than PB-IM and continues to widen their performance gap as the size of the seed set increases. Specifically, Hybrid-IM is about 43, 30, 4, and 40 times faster than PB-IM in NetHEPT, NetPHY, Stanford, and DBLP, respectively. Surprisingly, Hybrid-IM even outperforms SDD by 2.17 times in DBLP. It selects the seed set faster than SDD in DBLP when the size of the seed set is larger than or equal to 500.

Q4.2. Influence spread

In this experiment, we examined the influence spread of a seed set obtained by our Hybrid-IM, in comparison with that by existing IM methods used in Q4.1. We made each method select 1000 seed nodes and evaluated their influence spread by running MC-simulations. Fig. 10 shows the influence spread of the seed set by each method where the *x*-axis represents the number of seeds and the *y*-axis represents the influence spread.

CB-IM finds a seed set whose influence spread is lower than that obtained by Hybrid-IM by 26% and 30% in NetHEPT and NetPHY, respectively. This low quality of CB-IM's seed sets is due to the community detection using an *over-simplified graph*. This result shows that our PB-CD is superior in community detection to the original one in CB-IM.

SDD shows 22.9%, 33%, 40.6%, and 5.2% influence spread lower than our Hybrid-IM on the four datasets. SDD is a heuristic method which selects a node having the highest degree as a seed in a step. Once a seed is selected, SDD reduces the degree of all the nodes connected to the seed by 1 for the next node selection. This is intended to mitigate the problem of the

overlapping influence spread of neighbouring nodes to be computed more than once. In the three datasets, other than the Stanford dataset, SDD selects a seed set not only in a very short time but also with influence spread comparable to Hybrid-IM and PB-IM. The result is due to the characteristics of those datasets. They represent co-author relationships among authors where the degree of a node is low since an author tends to have a small number of her/his co-authors. As a result, the effect of decreasing the degree of neighbouring nodes of the seed appears significant. On the other hand, in the Stanford dataset with the max-degree of 38,606, the effect of decreasing the degree of a node by 1 is insignificant, which causes the influence spread of the seed set selected by SDD to be lower than the other methods. As a result, SDD showed the worst result in the Stanford dataset.

Finally, Hybrid-IM shows almost same influnce spread to CELFGreedy and PB-IM when the size of the seed set is under 100. With the increasing seed set size, Hybrid-IM shows 3.73% lower influence spread on average than PB-IM. In more detail, it shows 96%, 95.5%, 93.9%, and 99.4% of the influence spread obtained by PB-IM in NetHEPT, NetPHY, Stanford, and DBLP, respectively. This (small) loss of influence spread comes from the community based selections. On the other hand, Hybrid-IM improves the running time of PB-IM greatly up to 43 times. This result demonstrates that the very small loss caused by our community detection method contributes to great improvement in performance

Therefore, SDD can be a good alternative only when the node degree in the dataset is low and the time for seed selection is crucial. PB-IM and CELFGreedy could be considered more suitable to IM than Hybrid-IM when we want a seed set having the higher influence spread, given with sufficient time for seed selection. The proposed Hybrid-IM could be a good choice when people want a seed set having high influence spread (i.e., very little loss) with a much smaller execution time, which will be typical in real-world applications where a time constraint exists.

5. Conclusions

In this paper, we proposed a hybrid approach, Hybrid-IM that combines PB-IM and CB-IM, in order to resolve the micro and macro level issues together in the IM problem. To refine it more, we identified two additional issues and proposed two strategies that address them. The first one is the PB-CD strategy that considers influence propagation more accurately in community detection. The second one is the G-CELF strategy that further optimizes seed selections from multiple communities without any sacrifice of accuracy. Through extensive experiments on four real-world datasets, we demonstrated that (1) our strategies are all effective as a component and that (2) Hybrid-IM selects a seed set that provides the comparable influence spread to the best state-of-the-art methods but achieves great improvement in running time over them.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP; Ministry of Science, ICT & Future Planning) (No. NRF-2017R1A2B3004581).

References

[1] A. Arora, S. Galhotra, S. Ranu, Debunking the myths of influence maximization: an in-depth benchmarking study, in: Proceedings of the ACM International Conference on Management of Data, ACM, 2017, pp. 651–666.
[2] A. Bozorgi, H. Haghighi, M.S. Zahedi, M. Rezvani, Incim: a community-based algorithm for influence maximization problem under the linear threshold model, Inf. Process. Manag. 52 (6) (2016) 1188–1199.
[3] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, J. Tang, Online topic-aware influence maximization, in: Proceedings of the VLDB Endowment 8(6) (2015) 666–677.
[4] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: Proceedings of the Fifteenth International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 199–208.
[5] W. Chen, Y. Yuan, L. Zhang, Scalable influence maximization in social networks under the linear threshold model, in: Proceedings of the Tenth International Conference on Data Mining, IEEE, 2010, pp. 88–97.
[6] P. Domingos, M. Richardson, Mining the network value of customers, in: Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 57–66.
[7] S. Galhotra, A. Arora, S. Virinchi, S. Roy, Asim: a scalable algorithm for influence maximization under the independent cascade model, in: Proceedings of the Twenty-Fourth International Conference on World Wide Web, ACM, 2015, pp. 35–36.
[8] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[9] M. Gong, J. Yan, B. Shen, L. Ma, Q. Cai, Influence maximization in social networks based on discrete particle swarm optimization, Inf. Sci. (Ny) 367 (2016) 600–614.
[10] A. Goyal, F. Bonchi, L.V. Lakshmanan, Learning influence probabilities in social networks, in: Proceedings of the Third International Conference on Web Search and Data Mining, ACM, 2010, pp. 241–250.
[11] A. Goyal, F. Bonchi, L.V. Lakshmanan, A data-based approach to social influence maximization, in: Proceedings of the VLDB Endowment 5(1) (2011) 73–84.
[12] A. Goyal, W. Lu, L.V. Lakshmanan, Simpath: an efficient algorithm for influence maximization under the linear threshold model, in: Proceedings of the Eleventh International Conference on Data Mining, IEEE, 2011, pp. 211–220.
[13] M. Hosseini-Pozveh, K. Zamanifar, A.R. Naghsh-Nilchi, A community-based approach to identify the most influential nodes in social networks, J. Inf. Sci. 43 (2) (2017) 204–220.
[14] K. Jung, W. Heo, W. Chen, Irie: Scalable and robust influence maximization in social networks, in: Proceedings of the Twelfth International Conference on Data Mining, IEEE, 2012, pp. 918–923.
[15] D. Kempe, J. Kleinberg, É. Tardos, Maximizing the spread of influence through a social network, in: Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 137–146.
[16] J. Kim, S.-K. Kim, H. Yu, Scalable and parallelizable processing of influence maximization for large-scale social networks, in: Proceedings of the Twenty-Ninth International Conference on Data Engineering, IEEE, 2013, pp. 266–277.

[17] S. Kim, D. Kim, J. Oh, J.-H. Hwang, W.-S. Han, W. Chen, H. Yu, Scalable and parallelizable influence maximization with random walk ranking and rank merge pruning, Inf. Sci. (Ny) 415 (2017) 171–189.
[18] M. Kimura, K. Saito, Tractable models for information diffusion in social networks, in: Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2006, pp. 259–271.
[19] Y.-S. Kwon, S.-W. Kim, S. Park, An analysis of information diffusion in the blog world, in: Proceedings of the First International Workshop On Complex Networks Meet Information & Knowledge Management, ACM, 2009, pp. 27–30.
[20] Y.-S. Kwon, S.-W. Kim, S. Park, S.-H. Lim, J.B. Lee, The information diffusion model in the blog world, in: Proceedings of the Third Workshop on Social Network Mining and Analysis, ACM, 2009, p. 4.
[21] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: Proceedings of the Thirteenth International Conference on Knowledge Discovery and Data Mining, ACM, 2007, pp. 420–429.
[22] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters, Internet Math. 6 (1) (2009) 29–123.
[23] S.-H. Lim, S.-W. Kim, S. Park, J.H. Lee, Determining content power users in a blog network: an approach and its applications, IEEE Trans. Syst. Man Cyber. Part A Syst. Hum. 41 (5) (2011) 853–862.
[24] H. Ma, H. Yang, M.R. Lyu, I. King, Mining social networks using heat diffusion processes for marketing candidates selection, in: Proceedings of the Seventeenth International Conference on Information and knowledge Management, ACM, 2008, pp. 233–242.
[25] S. Peng, A. Yang, L. Cao, S. Yu, D. Xie, Social influence modeling using information theory in mobile social networks, Inf. Sci. (Ny) 379 (2017) 146–159.
[26] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.
[27] M. Richardson, P. Domingos, Mining knowledge-sharing sites for viral marketing, in: Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining, ACM, 2002, pp. 61–70.
[28] K. Saito, R. Nakano, M. Kimura, Prediction of information diffusion probabilities for independent cascade model, in: Proceedings of the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, Springer, 2008, pp. 67–75.
[29] G. Song, X. Zhou, Y. Wang, K. Xie, Influence maximization on large-scale mobile social network: a divide-and-conquer method, IEEE Trans. Parallel Distrib. Syst. 26 (5) (2015) 1379–1392.
[30] L.G. Valiant, The complexity of enumeration and reliability problems, SIAM J. Comput. 8 (3) (1979) 410–421.
[31] Y. Wang, G. Cong, G. Song, K. Xie, Community-based greedy algorithm for mining top-K influential nodes in mobile social networks, in: Proceedings of the Sixteenth International Conference on Knowledge Discovery and Data Mining, ACM, 2010, pp. 1039–1048.
[32] J. Weng, E.-P. Lim, J. Jiang, Q. He, Twitterrank: finding topic-sensitive influential twitterers, in: Proceedings of the Third International Conference on Web search and Data Mining, ACM, 2010, pp. 261–270.
[33] W. Xu, W. Liang, X. Lin, J.X. Yu, Finding top-k influential users in social networks under the structural diversity model, Inf. Sci. (Ny) 355 (2016) 110–126.
[34] A. Yadav, B. Wilder, E. Rice, R. Petering, J. Craddock, A. Yoshioka-Maxwell, M. Hemler, L. Onasch-Vera, M. Tambe, D. Woo, Influence maximization in the field: The arduous journey from emerging to deployed application, in: Proceedings of the Sixteenth Conference on Autonomous Agents and Multi Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2017, pp. 150–158.
[35] S.-H. Yoon, J.-H. Shin, S.-W. Kim, S. Park, Extraction of a latent blog community based on subject, in: Proceedings of the Eighteenth ACM Conference on Information and Knowledge Management, ACM, 2009, pp. 1529–1532.
[36] T. Zhu, B. Wang, B. Wu, C. Zhu, Maximizing the spread of influence ranking in social networks, Inf. Sci. (Ny) 278 (2014) 535–544.